AgroSense API Documentation

1. Arquitetura

A **AgroSense API** utiliza uma arquitetura baseada no padrão **Camadas (Layered Architecture)**, que organiza o código de forma a separar responsabilidades, melhorar a manutenibilidade e garantir escalabilidade. A API segue os seguintes princípios:

- **Apresentação (Presentation Layer)**: Responsável por expor os endpoints da API e receber as requisições HTTP. Essa camada é composta pelos **Controllers**.
- **Aplicação (Application Layer)**: Esta camada contém a lógica de negócio, como regras e operações da aplicação. É onde as requisições dos controllers são processadas e repassadas à camada de dados.
- **Dados (Data Layer)**: Encapsula a lógica de persistência e a comunicação com o banco de dados Oracle. Utiliza o **Entity Framework Core** e o **Oracle Managed Data Access** para a comunicação com o banco de dados.

2. Design Patterns Utilizados

A API utiliza alguns padrões de projeto para garantir que o código seja mais robusto e fácil de manter:

- **Repository Pattern**: Esse padrão é utilizado para encapsular a lógica de acesso aos dados e promover uma separação clara entre as camadas de aplicação e de dados. Isso facilita a substituição ou extensão do banco de dados no futuro sem impactar a lógica de negócios.
- **Dependency Injection (DI)**: O ASP.NET Core utiliza DI para injetar dependências necessárias nas classes. A injeção de dependência é usada para registrar e fornecer os serviços necessários, como o `DbContext`.
- **Singleton Pattern**: Utilizado para garantir que apenas uma instância de certos objetos críticos, como o `DbContext` e o `OracleConnection`, seja criada durante o ciclo de vida da aplicação.

3. Endpoints CRUD

A API expõe endpoints CRUD (Create, Read, Update, Delete) para gerenciar os recursos **Clientes** e **Vegetais**. Aqui está um exemplo básico dos endpoints implementados:

- **GET /api/vegetais**: Retorna todos os vegetais cadastrados.
- **GET /api/vegetais/{id}**: Retorna um vegetal específico pelo seu ID.
- **POST /api/vegetais**: Adiciona um novo vegetal.
- **PUT /api/vegetais/{id}**: Atualiza um vegetal existente.
- **DELETE /api/vegetais/{id}**: Exclui um vegetal específico.

Esses endpoints também estão disponíveis para o recurso **Clientes**, seguindo a mesma estrutura.

4. Documentação e Swagger

A documentação da API é feita através do **Swagger**. O Swagger UI permite que os desenvolvedores interajam com os endpoints da API diretamente no navegador e testem as funcionalidades.

Para acessar a documentação interativa, após rodar a API, acesse:

```
https://localhost:{porta}/swagger/index.html
```

5. Instruções para Rodar a API

Pré-requisitos

1. **.NET 8.0 SDK**: Instale o SDK do .NET 8.0 aqui.
2. **Oracle Database**: Configure o Oracle Database com as credenciais e permissões necessárias para a API.
3. **Editor de Código**: Utilize um IDE como **Visual Studio 2022** ou **Visual Studio Code**.

Passos para Rodar a API

1. **Clone o Repositório**:

   ```
   git clone https://github.com/seu-usuario/AgroSenseAPI.git
   cd AgroSenseAPI
   ```

2. **Configurar a String de Conexão**: Edite o arquivo **`appsettings.json`** para adicionar a string de conexão com o banco de dados Oracle:

   ```
   json
   Copiar código
   "ConnectionStrings": {
     "DefaultConnection": "Data Source=your-database-source;User
   Id=your-username;Password=your-password;"
   }
   ```

3. **Restaurar Dependências**: No diretório raiz do projeto, execute:

   ```
   dotnet restore
   ```

4. **Compilar e Rodar a Aplicação**:
   - No terminal:

     ```
     dotnet build
     dotnet run
     ```

   - Alternativamente, no **Visual Studio**:
     - Clique em **F5** para rodar a aplicação em modo de depuração.
5. **Acessar o Swagger UI**: Acesse o Swagger UI no navegador para testar os endpoints:

   ```
   https://localhost:{porta}/swagger/index.html
   ```

Ambiente de Produção

Para rodar a API em produção, configure o ambiente de hospedagem e use as ferramentas de publicação adequadas do **Azure**, **AWS**, ou **Docker**, conforme necessário.

Com essas informações, você está pronto para rodar a **AgroSense API** e utilizar os recursos CRUD de Clientes e Vegetais. Se algum dos endpoints CRUD não estiver funcionando corretamente, corrija os erros antes da entrega para evitar penalizações.